Streamline Integration using MPI-Hybrid Parallelism on a Large Multi-Core Architecture

David Camp (LBL, UC Davis), Hank Childs (LBL, UC Davis), Christoph Garth (UC Davis), Dave Pugmire (ORNL), & Kenneth I. Joy (UC Davis)
Computer Architecture

- CPUs
  - Single core CPUs
  - Multi-core CPUs
- Supercomputers
  - Thousands of nodes
    - Each node contains one or more multi-core CPUs
  - What code models should we be using for these systems?
Hybrid Parallelism Blends Distributed- and Shared-Memory Concepts

- **MPI-Only Model**
  - One MPI task per core
  - Nothing is shared between tasks

- **MPI-Hybrid Model**
  - One MPI task per node
  - Everything is shared between cores
Related Work in Hybrid Parallelism

- Relatively new area of research – hot topic in HPC
- Studies focus on “solvers,” not vis/graphics.
  - Parallel visualization applications all use MPI, none “multi-core” aware.
- Fundamental design questions:
  - What is the right balance of distributed- vs. shared-memory parallelism?
  - What memory patterns should we use?
  - What communication patterns should we use?
  - How do these factors impact performance?
Related Work in Hybrid Parallelism

• Conclusions of previous works in hybrid parallelism:
  • Answer: it depends on a complex set of factors.

• Many factors influence performance/scalability:
  • Synchronization overhead.
  • Load balance (intra- and inter-chip).
  • Communication patterns and overhead costs.
  • Memory access patterns.
  • Number of runtime threads.
Hybrid Parallelism for Streamline Integration on Large, Multi-core Platforms

• Our problem
  • Establish additional evidence as to the importance of hybrid parallelism for visualization and analysis
    • Is it better? Why and how much?

• Our experiments shows performance gains up to 10x speed up.
We Designed Experiments to Measure the Benefits of Hybrid Parallelism

- Streamlines have many characteristics that impact run time:
  - Data set size
  - Number of seed points
  - Seed point locations
  - Vector field complexity
- Our experiment were designed to cover these characteristics.
- We ran 48 tests. More in next slide.
- Tests ran on Franklin, a Cray XT4
  - MPI-Only tests used 32 nodes / 128 MPI tasks
  - MPI-Hybrid tests used 32 nodes / 32 MPI tasks
We Designed Experiments to Measure the Benefits of Hybrid Parallelism

- Sparse vs. Dense seed point placement
  - Seed points randomly throughout entire data set
  - Seed points located on area of interest
- Long vs. Short integration time
  - 4,000 and 1,000 time units
- Number of Seed Points
  - 2,500 and 10,000 seed points
- Three different data sets to explore vector field complexity
- Two different parallelization strategies
  - Parallelize-over-Seeds & Parallelize-over-Data
Parallelize-over-Seeds

- Example: 2d data set with 9 data blocks and 3 seed points partitioned over 3 tasks
Hybrid Parallelism for Parallelize-over-Seeds

- **Expected benefits:**
  - Larger cache allows for reduced reads.
  - Only need to read blocks once per node, instead of once per core.
Measuring the Benefits of Hybrid Parallelism for Parallelize-over-Seeds

![Graph showing improvement in MPI-Hybrid over MPI-only for different domains: Astrophysics, Thermal Hydraulics, and Fusion. The graph compares the performance of MPI-only and MPI-hybrid methods, with Integration Ratio indicated by dashed lines.](image-url)
Parallelize-over-Seeds

Integration Ratio

Blocks Loaded

MPI-only

MPI-hybrid

A(LD)  A(LS)  A(SD)  A(SS)  T(LD)  T(LS)  T(SD)  T(SS)  F(LD)  F(LS)  F(SD)  F(SS)
Gantt Chart for Parallelize-over-Seeds

MPI-Hybrid

MPI-Only

Time

Time

I/O
Integration
Parallelize-over-Data

- Example: 2d data set with 9 data blocks partitioned over 3 tasks
Hybrid Parallelism for Parallelize-over-Data

- **Expected benefits:**
  - Less communication
  - Should be able to reduce starvation by sharing data with a group of cores
Measuring the Benefits of Hybrid Parallelism for Parallelize-over-Data

![Graph showing improvement in MPI-Hybrid over MPI-only for different applications with specific labels for each category: Astrophysics, Thermal Hydraulics, Fusion. The graph includes columns for A(LD), A(LS), A(SD), A(SS), T(LD), T(LS), T(SD), T(SS), F(LD), F(LS), F(SD), F(SS), with bars indicating the performance improvement with colors for MPI-only and MPI-hybrid.]
Parallelize-over-Data

Integration Ratio

Bytes Communicated

MPI-only

MPI-hybrid
Gantt Chart for Parallelize-over-Data

MPI-Hybrid
8 cores integrating

MPI-Only
2 cores integrating

VisWeek 2011
### TABLE I
RESULTS FOR THE PARALLELIZE-OVER-SEEDS ALGORITHM

<table>
<thead>
<tr>
<th>Test case</th>
<th>$T_{\text{total}}$</th>
<th>$N_{\text{load}}$</th>
<th>$N_{\text{purged}}$</th>
<th>$T_{\text{I/O}}$</th>
<th>$T_{\text{int}}$</th>
<th>$R_{\text{int}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A(LD)</td>
<td>12.5s</td>
<td>12.5s</td>
<td>5,568</td>
<td>2,185</td>
<td>1,832</td>
<td>0</td>
</tr>
<tr>
<td>A(LS)</td>
<td>41.2s</td>
<td>19.3</td>
<td>8,405</td>
<td>3,525</td>
<td>4,573</td>
<td>17.5</td>
</tr>
<tr>
<td>A(SD)</td>
<td>62.1s</td>
<td>14.6s</td>
<td>6,699</td>
<td>2,914</td>
<td>2,995</td>
<td>2</td>
</tr>
<tr>
<td>A(SS)</td>
<td>63.4s</td>
<td>19.7s</td>
<td>9,787</td>
<td>4,151</td>
<td>5,986</td>
<td>493</td>
</tr>
<tr>
<td>T(LD)</td>
<td>27.3s</td>
<td>9.3s</td>
<td>6,544</td>
<td>2,194</td>
<td>2,710</td>
<td>0</td>
</tr>
<tr>
<td>T(LS)</td>
<td>43.5s</td>
<td>12.3s</td>
<td>4,535</td>
<td>2,755</td>
<td>1,130</td>
<td>2</td>
</tr>
<tr>
<td>T(SD)</td>
<td>53.6s</td>
<td>22.0s</td>
<td>12,715</td>
<td>5,153</td>
<td>8,875</td>
<td>1,313</td>
</tr>
<tr>
<td>T(SS)</td>
<td>59.9s</td>
<td>24.8</td>
<td>9,776</td>
<td>5,449</td>
<td>5,969</td>
<td>1,610</td>
</tr>
<tr>
<td>F(LD)</td>
<td>43.2s</td>
<td>13.0s</td>
<td>6,621</td>
<td>1,461</td>
<td>2,781</td>
<td>0</td>
</tr>
<tr>
<td>F(LS)</td>
<td>360s</td>
<td>36.1s</td>
<td>44,864</td>
<td>29,917</td>
<td>41,024</td>
<td>3,533</td>
</tr>
<tr>
<td>F(SD)</td>
<td>381s</td>
<td>39.9s</td>
<td>74,487</td>
<td>5,343</td>
<td>70,647</td>
<td>1,726</td>
</tr>
<tr>
<td>F(SS)</td>
<td>717s</td>
<td>125s</td>
<td>117,672</td>
<td>30,524</td>
<td>113,842</td>
<td>26,684</td>
</tr>
</tbody>
</table>

### TABLE II
RESULTS FOR THE PARALLELIZE-OVER-BLOCKS ALGORITHM

<table>
<thead>
<tr>
<th>Test case</th>
<th>$T_{\text{total}}$</th>
<th>$N_{\text{comm}}$</th>
<th>$T_{\text{comm}}$</th>
<th>$T_{\text{int}}$</th>
<th>$R_{\text{int}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A(LD)</td>
<td>21.2s</td>
<td>10.95s</td>
<td>26.69MB</td>
<td>11.96MB</td>
<td>2,006s</td>
</tr>
<tr>
<td>A(LS)</td>
<td>13.8s</td>
<td>5.76s</td>
<td>29.63MB</td>
<td>11.08MB</td>
<td>894s</td>
</tr>
<tr>
<td>A(SD)</td>
<td>8.7s</td>
<td>3.50s</td>
<td>9.76MB</td>
<td>4.82MB</td>
<td>690s</td>
</tr>
<tr>
<td>A(SS)</td>
<td>6.9s</td>
<td>3.39s</td>
<td>11.75MB</td>
<td>5.18MB</td>
<td>493s</td>
</tr>
<tr>
<td>T(LD)</td>
<td>25.3s</td>
<td>4.71s</td>
<td>22.68MB</td>
<td>12.3MB</td>
<td>990s</td>
</tr>
<tr>
<td>T(LS)</td>
<td>3.3s</td>
<td>0.79s</td>
<td>11.26MB</td>
<td>3.75MB</td>
<td>104s</td>
</tr>
<tr>
<td>T(SD)</td>
<td>5.6s</td>
<td>1.14s</td>
<td>7.37MB</td>
<td>4.3MB</td>
<td>230s</td>
</tr>
<tr>
<td>T(SS)</td>
<td>3.2s</td>
<td>0.59s</td>
<td>5.62MB</td>
<td>2.86MB</td>
<td>71s</td>
</tr>
<tr>
<td>F(LD)</td>
<td>36.4s</td>
<td>33.20s</td>
<td>52.51MB</td>
<td>26.92MB</td>
<td>3,789s</td>
</tr>
<tr>
<td>F(LS)</td>
<td>6.0s</td>
<td>3.57s</td>
<td>44.77MB</td>
<td>20.11MB</td>
<td>326s</td>
</tr>
<tr>
<td>F(SD)</td>
<td>22.8s</td>
<td>11.10s</td>
<td>46.08MB</td>
<td>23.63MB</td>
<td>2,197s</td>
</tr>
<tr>
<td>F(SS)</td>
<td>5.1s</td>
<td>3.13s</td>
<td>31.53MB</td>
<td>15.77MB</td>
<td>268s</td>
</tr>
</tbody>
</table>
Conclusions
Parallelize-over-Seeds

- MPI-Hybrid
  - Improved Performance
    - 2x – 10x speed-up
  - Better memory usage
  - Reduce I/O usage
    - By reducing # of data loads
  - Improved integration ratio
  - Better load balance
Conclusions
Parallelize-over-Data

- MPI-Hybrid far superior
  - Improved Performance
    - 2x – 5x speed-up
  - Significantly fewer messages and less data communicated
  - Improved integration ratio
  - Better load balance
Conclusions

• Benefits of a hybrid parallel approach
  • Improved performance
    • Uses less communication
    • Reduces memory redundancy
    • Less I/O bandwidth
    • Better load balancing
  • Observed speedups ranging from 2X to 10X
Thank you for your attention!!!

Streamline Integration using MPI-Hybrid Parallelism on a Large Multi-Core Architecture

David Camp (LBL, UC Davis), Hank Childs (LBL, UC Davis), Christoph Garth (UC Davis), Dave Pugmire (ORNL), & Kenneth I. Joy (UC Davis)